



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/833,248	04/11/2001	Robert Hundt	10005461-1	3718

7590 05/03/2004
HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P.O. Box 272400
Fort Collins, CO 80527-2400

EXAMINER

INGBERG, TODD D

ART UNIT	PAPER NUMBER
----------	--------------

2124

DATE MAILED: 05/03/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary

Application No.

09/833,248

Applicant(s)

HUNDT ET AL.

Examiner

Todd Ingberg

Art Unit

2124

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 09 February 2004.
- 2a) ☒ This action is **FINAL**. 2b) ☐ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☒ Claim(s) 1 and 3-17 is/are pending in the application.
- 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) ☐ Claim(s) _____ is/are allowed.
- 6) ☐ Claim(s) 1, 3-17 is/are rejected.
- 7) ☐ Claim(s) _____ is/are objected to.
- 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☐ The drawing(s) filed on _____ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some * c) ☐ None of:
- ☐ Certified copies of the priority documents have been received.
 - ☐ Certified copies of the priority documents have been received in Application No. _____.
 - ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- 1) ☒ Notice of References Cited (PTO-892)
- 2) ☐ Notice of Draftperson's Patent Drawing Review (PTO-948)
- 3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
Paper No(s)/Mail Date 4.
- 4) ☐ Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____.
- 5) ☐ Notice of Informal Patent Application (PTO-152)
- 6) ☐ Other: _____.

Claims 1 , 3 - 17 have been examined.

Claim 2 has been canceled.

Claims 1, 3 and 15 have been amended.

Claims 16 and 17 have been added.

Information Disclosure Statement

1. The Information Disclosure Statement filed on January 20, 2004 has been considered.

Interpretations

2. The following are the interpretations of the Examiner during the prosecution of the case.
 - a. **Functions** – The Specification states they comprise of an entry point and an endpoint. The Examiner notes this reads on the definition of a “Basic Block”.
 - b. **Instrumentation code** – the Specification describes this as what is commonly known as the ability to profile. However, many techniques for enabling profiling exist the use of adding pointers to instrumentation routines is the implementation.
 - c. **“encountering the branch instruction”** – is the call of a basic block.
 - d. **Substitute Version** – running a module other than the original basic block or running an altered basic block.

Art Unit: 2124

Claim Rejections - 35 USC § 102

3. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

The changes made to 35 U.S.C. 102(e) by the American Inventors Protection Act of 1999 (AIPA) and the Intellectual Property and High Technology Technical Amendments Act of 2002 **do not apply when the reference is a U.S. patent** resulting directly or indirectly from an international application **filed before November 29, 2000**. Therefore, the prior art date of the reference is determined under **35 U.S.C. 102(e) prior to the amendment by the AIPA** (pre-AIPA 35 U.S.C. 102(e)).

Claims 1 – 15 are rejected under 35 U.S.C. 102(e) as being anticipated by USPN # 6,189,141 **Benitez et al.**

The applied reference has a common Assignee (HP) with the instant application. Based upon the earlier effective U.S. filing date of the reference, it constitutes prior art under 35 U.S.C. 102(e).

	<u>Benitez et al.</u>	<u>09/833,248</u>
Filed Date	May 4, 1998	April 11, 2001
Issued Date	February 13, 2001	

Benitez filed before November 29, 2000.

Claim 1

Art Unit: 2124

Benitez anticipates a computer-implemented method for dynamic instrumentation of an executable application program (**Benitez**, Abstract – runtime based on dynamic evaluation of control flow) using an instrumentation program (**Benitez**, ability to instrument code to determine dynamically HOT and COLD activity) the application program including a plurality of original functions (**Benitez**, by computer executable programs inherently have “basic blocks”, This term is the term Benitez has adapted the term “Hot Block” to – col 2, lines 55 - 60), each original function having an entry point and an endpoint (**Benitez** – inherent for a basic block to have a start and end – col 55 – 64), comprising: patching the function entry points with breakpoint instructions; and creating the substitute functions upon encountering the breakpoint instructions. (**Benitez**, the “trace” placed in the block as stated col 2 lines 53 – 55 – this adds the instrumentation to the block after the trace is added the block is considered a substitute function) creating a shared memory segment for the instrumentation program and the application program (**Benitez** – col 19, lines 35 – 40 – ability to remove blocks when they get cold – also note figure 9 as noted below) ; upon initial invocation of the original functions in the application program (**Benitez** – col 19, lines 35 – 40 – ability to remove blocks when they get cold – also note figure 9 as noted below), and in response to encountering the breakpoint instructions. (**Benitez**, the “trace” placed in the block as stated col 2 lines 53 – 55 – this adds the instrumentation to the block after the trace is added the block is considered a substitute function) creating in the shared memory segment corresponding substitute functions including instrumentation code (**Benitez** – col 19, lines 35 – 40 – ability to remove blocks when they get cold – also note figure 9 as noted below) ; and executing the substitute functions (**Benitez** – Figure 9 shows the instrumented

Art Unit: 2124

blocks such as #224 – 930 to 932) in lieu of the original functions in the application program (Benitez – running the instrumented block).

Claim 2

Canceled

Claim 3

The method of claim 2, further comprising replacing the break instruction at the entry points of the functions in the application program with branch instructions that target the substitute functions. (Benitez, col 2 lines 60 – 65, the “arc” is a jump (JMP) instruction which is a branch)

Claim 4

The method of claim 3, wherein the executable application program includes one or more branch instructions having target addresses that reference entry points of one or more of the original functions, further comprising: after creating a substitute function corresponding to an original function, for a branch instruction that references the original function replacing the target addresses to reference the substitute function. (Benitez, col 2 lines 60 – 65, the “arc” is a JMP from block to block – claim 1 and 2 above cover the blocks having been instrumented – Another interpretation of the claim language is that the Optimizer orders the blocks which could be called a substitute function see col 32 lines 1 – 10 the dynamically optimizes intermediate representation is the order of the basic blocks also called the DAG.)

Claim 5

The method of claim 1, wherein the executable application program includes one or more branch instructions having target addresses that reference entry points of one or more of the original functions, further comprising: after creating a substitute function corresponding to an original

Art Unit: 2124

function, for a branch instruction that references the original function replacing the target addresses to reference the substitute function. (**Benitez**, col 2 lines 60 – 65, the “arc” is a JMP from block to block – claim 1 and 2 above cover the blocks having been instrumented – Another interpretation of the claim language is that the Optimizer orders the blocks which could be called a substitute function see col 32 lines 1 – 10 the dynamically optimizes intermediate representation is the order of the basic blocks also called the DAG.).

Claim 6

The method of claim 1, further comprising: copying a segment of the executable application program to selected area of memory by the instrumentation program; replacing the segment of the application program with code that allocates the shared memory by the instrumentation program; executing the code in the application program that allocates the shared memory segment; and restoring the segment of the executable application from the selected area of memory to the application program by the instrumentation program after the shared memory is allocated. (**Benitez**, col 12, lines 20 – 50 also see col 34 the “Hot Trace Memory Manager”).

Claim 7

The method of claim 6, further comprising: patching the function entry points with breakpoint instructions; and creating the substitute functions upon encountering the breakpoint instructions. (**Benitez**, as per the rejection for claim 2 and claim 3).

Claim 8

The method of claim 7, further comprising replacing the break instruction at the entry points of the functions in the application program with branch instructions that target the substitute functions. (**Benitez**, as per the rejection for claim 2 and claim 3).

Claim 9

The method of claim 8, wherein the executable application program includes one or more branch instructions having target addresses that reference entry points of one or more of the original functions, further comprising: after creating a substitute function corresponding to an original function, for a branch instruction that references the original function replacing the target addresses to reference the substitute function. (**Benitez**, as per the rejection for claim 2 , claim 3 and claim 6).

Claim 10

The method of claim 6, wherein the executable application program includes one or more branch instructions having target addresses that reference entry points of one or more of the original functions, further comprising: after creating a substitute function corresponding to an original function, for a branch instruction that references the original function replacing the target addresses to reference the substitute function. (**Benitez**, as per the rejection for claim 2 , claim 3 and claim 6).

Claim 11

The method of claim 6, wherein the executable application program includes a plurality of threads and further comprising: before the step of copying the segment of the executable application program suspending all threads of the executable application program, and selecting one of the suspended threads; and after replacing the segment of the executable application program with the code that allocates the shared memory, resuming execution of the one of the suspended threads at the code that allocates the shared memory. (**Benitez**, as per the rejection for claim 2 , claim 3 and claim 6).

Art Unit: 2124

Claim 12

The method of claim 11, further comprising: patching the function entry points with breakpoint instructions; and creating the substitute functions upon encountering the breakpoint instructions.

(**Benitez**, as per the rejection for claim 2 , claim 3 and claim 6).

Claim 13

The method of claim 12, further comprising replacing the break instruction at the entry points of the functions in the application program with branch instructions that target the substitute functions. (**Benitez**, as per the rejection for claim 2 and claim 3).

Claim 14

The method of claim 13, wherein the executable application program includes one or more branch instructions having target addresses that reference entry points of one or more of the original functions, further comprising: after creating a substitute function corresponding to an original function, for a branch instruction that references the original function replacing the target addresses to reference the substitute function. (**Benitez**, as per the rejection for claim 2 , claim 3 and claim 6).

Claim 15

Benitez anticipates an apparatus for dynamic instrumentation of an executable application program (**Benitez**, Abstract – runtime based on dynamic evaluation of control flow) by an instrumentation program (**Benitez**, ability to instrument code to determine dynamically HOT and COLD activity), the application program including a plurality of original functions (**Benitez**, by computer executable programs inherently have “basic blocks”, This term is the term Benitez has adapted the term “Hot Block” to – col 2, lines 55 - 60), each original function having an entry

Art Unit: 2124

point and an endpoint (**Benitez** – inherent for a basic block to have a start and end – col 55 – 64), comprising: means for patching the function entry points with breakpoint instructions; and creating the substitute functions upon encountering the breakpoint instructions. (**Benitez**, the “trace” placed in the block as stated col 2 lines 53 – 55 – this adds the instrumentation to the block after the trace is added the block is considered a substitute function) means for creating a shared memory segment for the instrumentation program (**Benitez** – col 19, lines 35 – 40 – ability to remove blocks when they get cold – also note figure 9 as noted below) and the application program (**Benitez** – Figure 9 shows the instrumented blocks such as #224 – 930 to 932); means for creating in the shared memory segment corresponding substitute functions including instrumentation code upon initial invocation of the original functions in the application program (**Benitez** – Figure 9 and ability to fetch and load in and out as taught by determinations of COLD and HOT ,col 19, lines 35 – 40); and in response to encountering the breakpoint instructions. (**Benitez**, the “trace” placed in the block as stated col 2 lines 53 – 55 – this adds the instrumentation to the block after the trace is added the block is considered a substitute function) and means for executing the substitute functions in lieu of the original functions in the application program (**Benitez** – running the instrumented block).

Claim 16

Benitez anticipates the method of claim 3, wherein the executable application program is stored in memory (**Benitez** – see figure 8 for instrumenting the block and the look up table also note the Abstract states the hot block is dynamically instrumented this requires the use of memory, Figure 3 shows the runtime environment with the instruction processor also see col 9 and 10 and figure 6A through figure 6C) and includes one or more branch instructions having associated target addresses (**Benitez** – Figure 8 shows the original address as well as the translated instruction address – one or more branches to the address of the block is technical terms for the ability to

Art Unit: 2124

call the routine from more than one location in the program) that reference entry points of one or more of the original functions in the memory (**Benitez** – Figure 8 shows the original function instrumented the entry point is the initial address of the instrumented hot block this enables the counters as shown in figure 6D which shows details of the JMP that calls the routines (plural as in multiple entries in the table)), further comprising: after creating a substitute function corresponding to an original function (**Benitez** – Figure 8 as per the above), for each branch instruction that references the original function in memory, replacing, in the application program stored in the memory (**Benitez** – the address look up at runtime as per the sections above), the associated target address with an address that references the substitute function (**Benitez** – Figure 8 as per the rejection above).

Claim 17

The method of claim 1, wherein the executable application program is stored in memory (**Benitez** – Figure 8 shows the original address as well as the translated instruction address – one or more branches to the address of the block is technical terms for the ability to call the routine from more than one location in the program) and includes one or more branch instructions having associated target addresses that reference entry points of one or more of the original functions in the memory (**Benitez** – Figure 8 shows the original function instrumented the entry point is the initial address of the instrumented hot block this enables the counters as shown in figure 6D which shows details of the JMP that calls the routines (plural as in multiple entries in the table)), further comprising: after creating a substitute function corresponding to an original function (**Benitez** – the address look up at runtime as per the sections above), for each branch instruction that references the original function in memory (**Benitez** – the address look up at runtime as per the sections above), replacing, in the application program stored in the memory (**Benitez** – Figure 8 as per the rejection above), the associated target address with an address that references the substitute function (**Benitez** – Figure 8 as per the rejection above).

Response to Arguments

4. The following statements are the Applicant's"

Applicant's Statement

"The Office Action fails to establish that claims 1- 15 are anticipated by US patent number 6,189,141 to Benitez et al. ("Benitez") under 35 USC §102(e). The rejection is traversed

Art Unit: 2124

because the Office Action fails to show that all the limitations of the claims are identically taught by Benitez.

The rejection of claim 1 is traversed. However, the rejection is now moot in view of the limitations of claim 2 that have been added to claim 1. The rejection also fails to show that Benitez teaches the limitations of now-canceled claim 2. The limitations of combined claims 1 and 2 include patching the function entry points with breakpoint instructions; creating a shared memory segment for the instrumentation program and the application program; upon initial invocation of the original functions in the application program and in response to encountering the breakpoint instructions, creating in the shared memory segment corresponding substitute functions including instrumentation code; and executing the substitute functions in lieu of the original functions in the application program.

The Office Action alleges that Benitez's col. 2, 11. 53-55 teaches patching the function entry points with breakpoint instructions and creating substitute functions upon encountering the breakpoint instructions. However, this cited section of Benitez teaches, "A trace typically is made up of one or more blocks of original instructions of an executable file, each of which from the selected area of memory to the application program by the instrumentation program after the shared memory is allocated. The Office Action fails to show that these limitations are identically taught by Benitez in the cited Benitez's col. 12, 11. 20-50 and col. 34's description of the Hot Trace Memory Manager."

Examiner's Response

Applicant's argument that the identical invention is not taught by Benitez is not persuasive. Although the Benitez reference teaches more than the disclosed invention by using thresholds the basic concepts of compiler theory and instrumentation (profiling) have altered to one of ordinary skill in the art. Yes, Benitez has words like "**hot block**" and "**cold block**". These instrumented blocks are instrumented basic blocks. **Basic block** being a well known term in the art of compiler theory. One of ordinary skill in the art must understand the underlying structure of a computer program and what is needed for a program to run at runtime to be of very basic skill in the art. Benitez teaches the elements of the claimed invention as arranged in the claims but this is not an *ipse dixit* test, i.e., identity of terminology is not required. In re Bond, 910 F. 2d 831, 15 USPQ2d 1566 (Fed Cir., 1990).

Any time the Applicant amends directly into the prior art and particularly the grossly old and well known techniques in the art the Examiner has concern the Applicant might not fully understand the art of record.

Examiner urges Applicant to reread columns 9 and 10 of the Benitez reference. The starting target instruction is the beginning address of the function (entry point – as you call it). The hot trace optimizer and instrumenter identify hot blocks by usage statistics see figure 6D and the counters. Also in Figure 6D the entry point is the Start instruction Address. Logically taking a step back from the details of compiler theory this makes sense because the point of incrementing code is to determine the frequency routines are run. The beginning of the routine is the point to place the counter because entering the routine means it was called. Placing a counter after instructions risks another call or branch (JMP at the lower level) and not counting the call to this routine. The Benitez reference teaches the limitations but uses different terminology.

Art Unit: 2124

In terms of seeing the instrumented code before and after see the sequence of Figures 6A, 6B and 6C. The code once it is a HOT Block it is instrumented. The Hot Block and Hot Trace Look-Up table of Figure 8 show the before and after (altered addresses) of the routine. What the Applicant is calling the breakpoint instructions is the JMP to a routine. One of ordinary skill in the art understands programs are broken into basic blocks at the compiler level for the run time and multiple basic blocks can make up a routine. This is inherent in computer programs and meets the argument; "A trace typically is made up of one or more blocks of original instructions of an executable file, each of which from the selected area of memory to the application program by the instrumentation program after the shared memory is allocated." One of ordinary skill in the art also knows program require memory to run.

Applicant's Statement

Benitez's col. 12, 11. 20-50 discusses correlating original instructions with translated instructions in a lookup table and using the lookup table to determine whether to transfer control to the translated instruction. Benitez's description of the Hot Trace Memory Manager in col. 34 indicates that this component stores optimized and instrumented hot trace into hot trace storage and selectively removes from the storage area those hot blocks that have become cold. These sections do not suggest the limitations of claim 6.

Claim 6 clearly indicates that the application program (not an instrumentation program or Hot Trace Memory Manager) allocates the shared memory used for instrumentation by way of a segment of the application program that is temporarily replaced with code that allocates memory. This temporary replacement of application program code with code that allocates the shared memory segment is not taught by the cited sections of Benitez, nor seen to be suggested elsewhere in Benitez. Therefore, the Office Action fails to show that Benitez teaches the limitations of claim 6, and the rejection should be withdrawn.

Examiner's Response

Claim 6 is as follows:

"The method of claim 1, further comprising: copying a segment of the executable application program to selected area of memory by the instrumentation program; replacing the segment of the application program with code that allocates the shared memory by the instrumentation program; executing the code in the application program that allocates the shared memory segment; and restoring the segment of the executable application from the selected area of memory to the application program by the instrumentation program after the shared memory is allocated. "

Claim 6 is dependent on claim 1. The rejection of 1 and 6 meet the claim limitations. The rejection for 1 with the limitations of 2 is responded to in detail in the arguments section above. This explains why the (Benitez, col 12, lines 20 – 50 also see col 34 the "Hot Trace Memory Manager") is appropriate for the limitations of claim 6.

Applicant's Statement

Claims 7-14 depend directly or indirectly from the claims discussed above, and therefore, are allowable for at least the reasons set forth above.

Examiner's Response

Arguments for claim independent not persuasive rejection for claims maintained.

Art Unit: 2124

Applicant's Statement

Claim 15 is amended and includes limitations similar to those of amended claim 1. Therefore, claim 15 is not anticipated by Benitez.

Examiner's Response

Arguments for claim 1 not persuasive rejection for claim 15 maintained.

Conclusion

5. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

US Patent Literature

A. USPN 5,491,808 Geist, Jr, Instruments code for the purpose of dynamically tracking memory resource allocation and deallocation. Geist uses the grossly old and well known technique of instrumenting code at the entry point. This technique is well established in the art. And is used for not only instrumentation but also for patching as shown in the next reference.

Non Patent Literature

B. IBM Technical Disclosure shows when you patch a routine you by pass it with the use of a JMP to a new routine and update the pointer in the jump table. The jump table control path of the program execution and avoids the old code and substitutes the new code. In this particular example the old code is left in place but is not called.

6. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event,

Art Unit: 2124

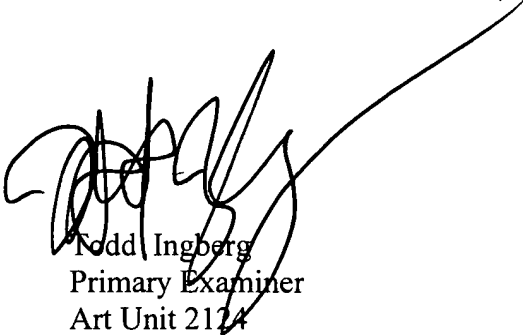
however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Correspondence Information

7. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Todd Ingberg whose telephone number is (703) 305-9775. The examiner can normally be reached on Monday - Thursday 6:30 AM to 5:00 PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (703) 305-9662. The fax phone number for the organization where this application or proceeding is assigned is (703) 746-7239.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-9700.



Todd Ingberg
Primary Examiner
Art Unit 2124

April 29, 2004